AMENDMENTS TO THE CLAIMS

1. (Currently amended) A method of managing communications between service components in a computing environment, the computing environment comprising an interconnection system, an entity external to the interconnection system and communicatively linked with the interconnection system, and a plurality of processing nodes interconnected by the interconnection system, each of the service components having a respective identity and being programmed on at least a respective one of the processing nodes, the method comprising:

based on the identity of at least one of the service components, establishing access-control logic restricting inter-node communication involving the at least one service component;

programming the external entity with the access-control logic;

at the external entity, receiving from the interconnection system a signal indicating detection of an attempted inter-node communication involving the at least one service component;

in response to receiving the signal, the external entity providing at least a portion of the access-control logic to the interconnection system ~~in response to an attempted inter-node communication involving the at least one service component~~; and

applying the access-control logic provided to the interconnection system, to block the attempted inter-node communication involving the at least one service component.

2. (Original) The method of claim 1, wherein establishing the access-control logic comprises:

establishing a rule indicating whether to allow a communication involving the at least one service component; and

translating the rule into the access-control logic.

3. (Original) The method of claim 2, wherein:

establishing a rule indicating whether to allow a communication involving the at least one service component comprises establishing a rule indicating whether to allow a communication with a first service component programmed on a first processing node in the computing environment.

4. (Original) The method of claim 3, wherein:

establishing a rule indicating whether to allow a communication involving the at least one service component comprises establishing a rule indicating whether to allow a communication between (i) a first service component programmed on a first processing node in the computing environment and (ii) a second service component programmed on a second processing node in the computing environment.

5. (Previously presented) The method of claim 3, wherein the at least one service component resides at at least one service-access-point in the computing environment, and wherein translating the rule into the access-control logic comprises mapping the rule into packet-filter logic associated with the at least one service-access-point.

6. (Original) The method of claim 5, wherein the at least one service-access-point comprises an IP address of the first processing node.

7. (Original) The method of claim 6, wherein the first processing node is programmed to associate a first transport port with the first service component, and wherein the at least one service-access-point further comprises the first transport port.

8. (Original) The method of claim 1, wherein applying the access-control logic to block an inter-node communication involving the at least one service component comprises:

detecting an attempted inter-node communication involving the at least one service component;

based on the access-control logic, making a determination that the attempted inter-node communication should be blocked; and

in response to the determination, blocking the attempted inter-node communication.

9. (Original) The method of claim 1, wherein at least two processing nodes of the plurality of interconnected processing nodes run different operating systems.

10. (Original) The method of claim 1, wherein at least two processing nodes of the plurality of interconnected processing nodes support different processor instructions sets.

11. (Original) The method of claim 1, wherein the computing environment is a cluster-based computing environment.

12. (Original) The method of claim 1, wherein the computing environment is a public computing platform.

13. (Currently amended) A method of managing communications between service components in a computing environment, the computing environment comprising an interconnection system, an entity external to the interconnection system and communicatively linked with the interconnection system, and a plurality of processing nodes interconnected by the interconnection system, each of the service components having a respective identity and being programmed on at least a respective one of the processing nodes, the method comprising:

based on the identity of at least one of the service components, establishing at least one access-control rule indicating whether to allow at least one communication involving the at least one service component;

translating the at least one access-control rule into access-control logic;

programming the external entity with the access-control logic;

~~detecting,~~ at the interconnection system, detecting an attempted inter-node communication between service components and responsively sending the external entity a signal indicating detection of the attempted inter-node communication;

at the external entity, receiving the signal indicating detection of the attempted inter-node communication and responsively providing at least a portion of the access-control logic to the interconnection system ~~in response to the attempted inter-node communication between service components~~;

based on the access-control logic provided to the interconnection system, determining that the attempted inter-node communication between service components is not allowed; and

responsively blocking the attempted inter-node communication.

14. (Previously presented) The method of claim 13, wherein the access-control logic comprises packet-filter logic.

15. (Previously presented) The method of claim 13, wherein:

each of the service components is designated by a respective service-access-point (SAP) in the computing environment, and each processing node has a respective SAP as well; and

the access-control logic comprises packet-filter logic associated with at least one SAP in the computing environment.

16. (Previously presented) The method of claim 15, wherein the respective SAP of each service component comprises an IP address of the respective processing node on which the service component is programmed, and wherein the access-control logic comprises packet-filter logic associated with at least one such IP address.

17. (Previously presented) The method of claim 16, wherein at least one of the SAPs of a service component further comprises a port selected from the group consisting of a TCP port and a UDP port, and wherein the packet-filter logic is further associated with at least one such port.

18. (Previously presented) The method of claim 13,

wherein the communications between service components are packet-based; and

wherein the access-control logic comprises packet-filter logic associated with a combination of at least (i) a packet transport protocol, (ii) a source address in the computing environment and (iii) a destination address in the computing environment.

19. (Previously presented) The method of claim 13, wherein translating the at least one access-control rule into access-control logic comprises:

mapping the at least one access-control rule to packet-filter logic associated with at least one service-access-point in the computing environment.

20. (Original) The method of claim 13,

wherein at least a given one of the processing nodes includes a firewall for restricting communications with the given processing node; and

translating the at least one access-control rule into access-control logic comprises provisioning the firewall of the given processing node to allow communications between at least one service component programmed on the given processing node and at least one service component programmed on another processing node.

21. (Previously presented) The method of claim 13, wherein the interconnection system further performs the element of blocking the attempted inter-node communication.

22-23. (Cancelled)

24. (Previously presented) The method of claim 13, further comprising:

providing at least another portion of the access-control logic to the interconnection system prior to detecting the attempted inter-node communication between service components.

25. (Cancelled)

26. (Previously presented) The method of claim 13, wherein the interconnection system comprises a switch, and wherein providing the at least a portion of the access-control logic to the interconnection system comprises setting up the switch to apply the access-control logic.

27. (Previously presented) The method of claim 26,

wherein the switch comprises (i) a packet-filtering agent and (ii) a provisioning-interface for receiving command-line instructions to set up the packet-filtering agent, the switch being arranged to translate the command-line instructions into packet-filtering logic executable by the packet-filtering agent; and

wherein setting up the switch to apply the access-control logic comprises providing the switch, via the provisioning-interface, with command-line instructions representative of the access-control logic.

28. (Previously presented) The method of claim 21, wherein an entity coupled to the interconnection system performs the element of providing at least a portion of the access-control logic to the interconnection system.

29. (Cancelled)

30. (Currently amended) The method of claim 28, wherein the external entity coupled to the interconnection system comprises a session manager.

31. (Original) The method of claim 21, wherein the interconnection system comprises a switch.

32. (Original) The method of claim 21, wherein the interconnection system comprises a router.

33. (Original) The method of claim 13, wherein the computing environment is a cluster-based computing environment.

34. (Previously presented) The method of claim 13, wherein the computing environment is a public computing platform.

35. (Original) The method of claim 13, wherein the attempted inter-node communication comprises an attempted inter-node communication between antagonistic service components.

36. (Original) The method of claim 13, wherein the attempted inter-node communication comprises an attempted communication of a packet from a first processing node to a second processing node, and wherein blocking the attempted communication comprises dropping the packet.

37. (Currently amended) A method for managing application logic in a public computing platform, the public computing platform comprising (i) a network of processing nodes interconnected by an interconnection system, and (ii) an entity external to the interconnection system and communicatively linked with the interconnection system, the method comprising:

receiving specifications of at least two computer-program applications, the applications cooperatively comprising a number of application components;

loading the application components of the at least two applications onto at least two of the processing nodes of the computing platform;

at the interconnection system, detecting an attempted inter-node communication between the application components and responsively sending to the external entity a signal indicating detection of the attempted inter-node communication;

at the external entity, receiving the signal indicating detection of the attempted inter-node communication and responsively providing to the interconnection system, ~~in response to an~~

~~attempted inter-node communication between the application components,~~ at least a portion of access-control rules that define allowed communications between the application components; and

applying the access-control rules provided to the interconnection system, to block the attempted inter-node communication between the application components.

38. (Currently amended) A computing environment with communication control comprising:

an interconnection system;

a plurality of co-located processing nodes interconnected via the interconnection system;

a plurality of application components loaded onto the processing nodes, each application component having a respective service-access-point defining location of the application component in the computing environment; and

an entity, external to the interconnection system, communicatively linked with the interconnection system,

wherein the external entity includes access-control logic indicating allowed inter-node communications between application components,[[;]]

wherein the interconnection system detects an attempted inter-node communication between application components and responsively sends to the external entity a signal indicating detection of the attempted inter-node communication,

wherein the external entity receives the signal and responsively provides the access-control logic to the interconnection system,

the access-control logic being executable, in response to the ~~an~~ attempted inter-node communication, to make a determination of whether the attempted inter-node communication is allowed; and

the access-control logic being executable, in response to a determination that the attempted inter-node communication is not allowed, to block the attempted inter-node communication; and

~~a session manager communicatively linked with the interconnection system, wherein the logic is located, at least in part, in the session manager, and wherein the session manager provides at least a portion of the logic to the interconnection system in response to the attempted inter-node communication~~.

39-41. (Cancelled)

42. (Original) The computing environment of claim 38, wherein the computing environment is a cluster-based computing environment.

43. (Original) The computing environment of claim 38, wherein the computing environment is a public-computing platform.

44. (Original) The method of claim 38, wherein at least two of the co-located processing nodes run different operating systems.

45. (Original) The method of claim 38, wherein at least two of the co-located processing nodes support different processor instructions sets.

46-49. (Cancelled)

50. (New) The method of claim 1, further comprising:

assigning to each service component a respective trustworthiness measure and a respective criticality measure,

using the trustworthiness and criticality measures of each service component to select the at least a respective one of the processing nodes onto which each service component should be programmed, and

programming each service component onto the at least a respective one of the processing nodes selected for the service component,

wherein the trustworthiness measure for each service component represents an assessment of a potential threat the service component poses to other objects, and

wherein the criticality measure for each service component represents a measure selected from the group consisting of: (i) a measure of importance of the service component, and (ii) a measure of concern for what the service component may do to other service components.

51. (New) The method of claim 13, further comprising:

assigning to each service component a respective trustworthiness measure and a respective criticality measure,

using the trustworthiness and criticality measures of each service component to select the at least a respective one of the processing nodes onto which each service component should be programmed, and

programming each service component onto the at least a respective one of the processing nodes selected for the service component,

wherein the trustworthiness measure for each service component represents an assessment of a potential threat the service component poses to other objects, and

wherein the criticality measure for each service component represents a measure selected from the group consisting of: (i) a measure of importance of the service component, and (ii) a measure of concern for what the service component may do to other service components.

52. (New) The method of claim 37, further comprising:

assigning to each application component a respective trustworthiness measure and a respective criticality measure,

using the trustworthiness and criticality measures of each application component to select the at least a respective one of the processing nodes onto which each application component should be programmed, and

programming each application component onto the at least a respective one of the processing nodes selected for the application component,

wherein the trustworthiness measure for each application component represents an assessment of a potential threat the application component poses to other objects, and

wherein the criticality measure for each application component represents a measure selected from the group consisting of: (i) a measure of importance of the application component,

and (ii) a measure of concern for what the application component may do to other application components.

53. (New) The computing environment of claim 38, further comprising:

assigning to each application component a respective trustworthiness measure and a respective criticality measure,

using the trustworthiness and criticality measures of each application component to select the at least a respective one of the processing nodes onto which each application component should be programmed, and

programming each application component onto the at least a respective one of the processing nodes selected for the application component,

wherein the trustworthiness measure for each application component represents an assessment of a potential threat the application component poses to other objects, and

wherein the criticality measure for each application component represents a measure selected from the group consisting of: (i) a measure of importance of the application component, and (ii) a measure of concern for what the application component may do to other application components.